

Qt



# Qt For Robots: Interfaces, Sensores e Simuladores

---

**Patrick Pereira**

[patrick@bluerobotics.com](mailto:patrick@bluerobotics.com)

[patrickjp@kde.org](mailto:patrickjp@kde.org)





# whoami?

Patrick Pereira / Electronic Eng.



@patrickelectric



@patrickelectric



@patrickelectric



<https://patrickelectric.work>



# Work



## Software Engineer

Software development for ROVs, sensors and user applications.

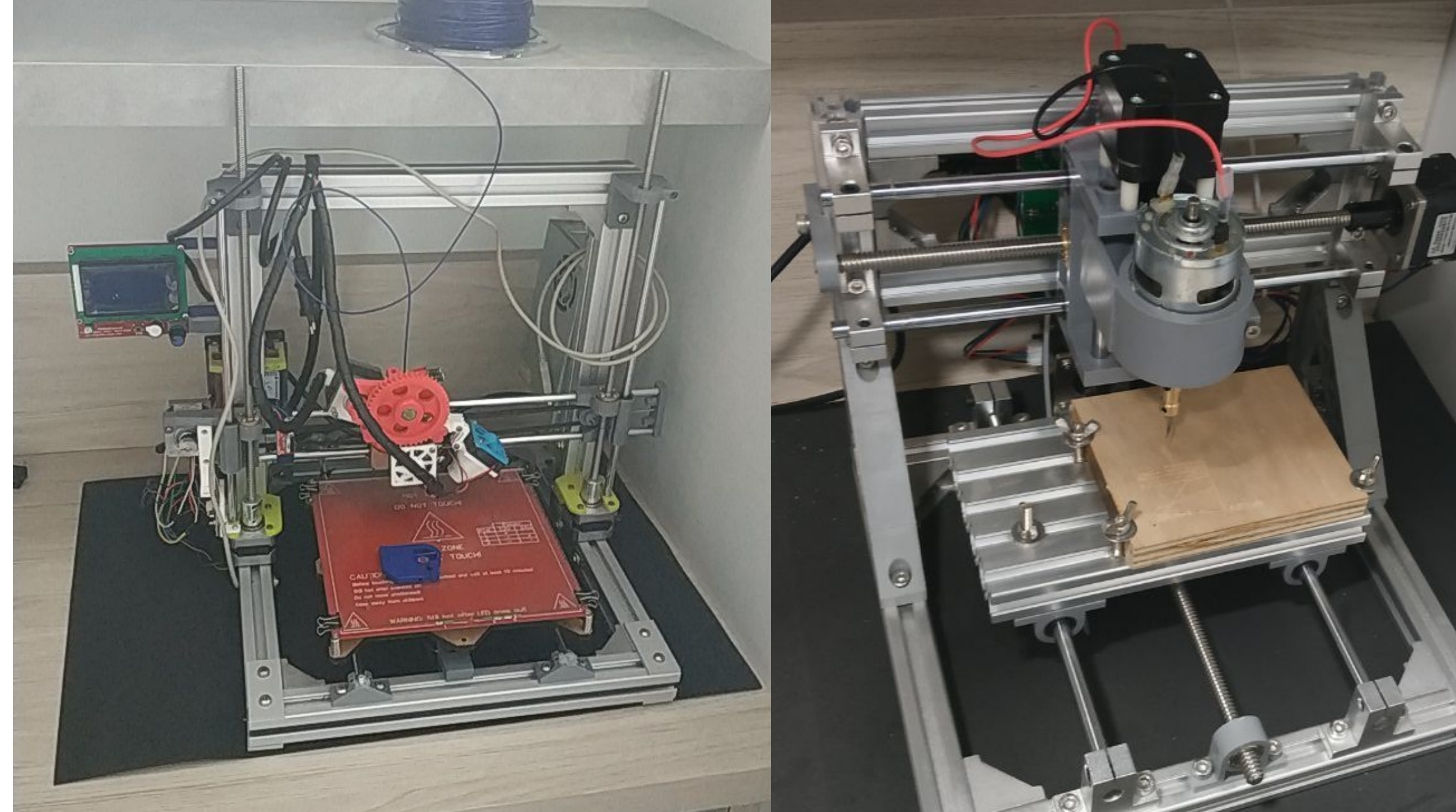
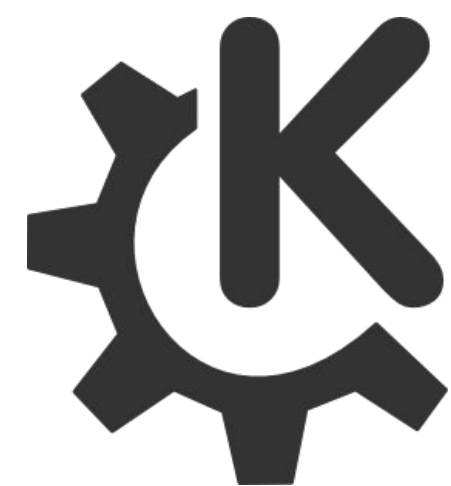


# Hobby



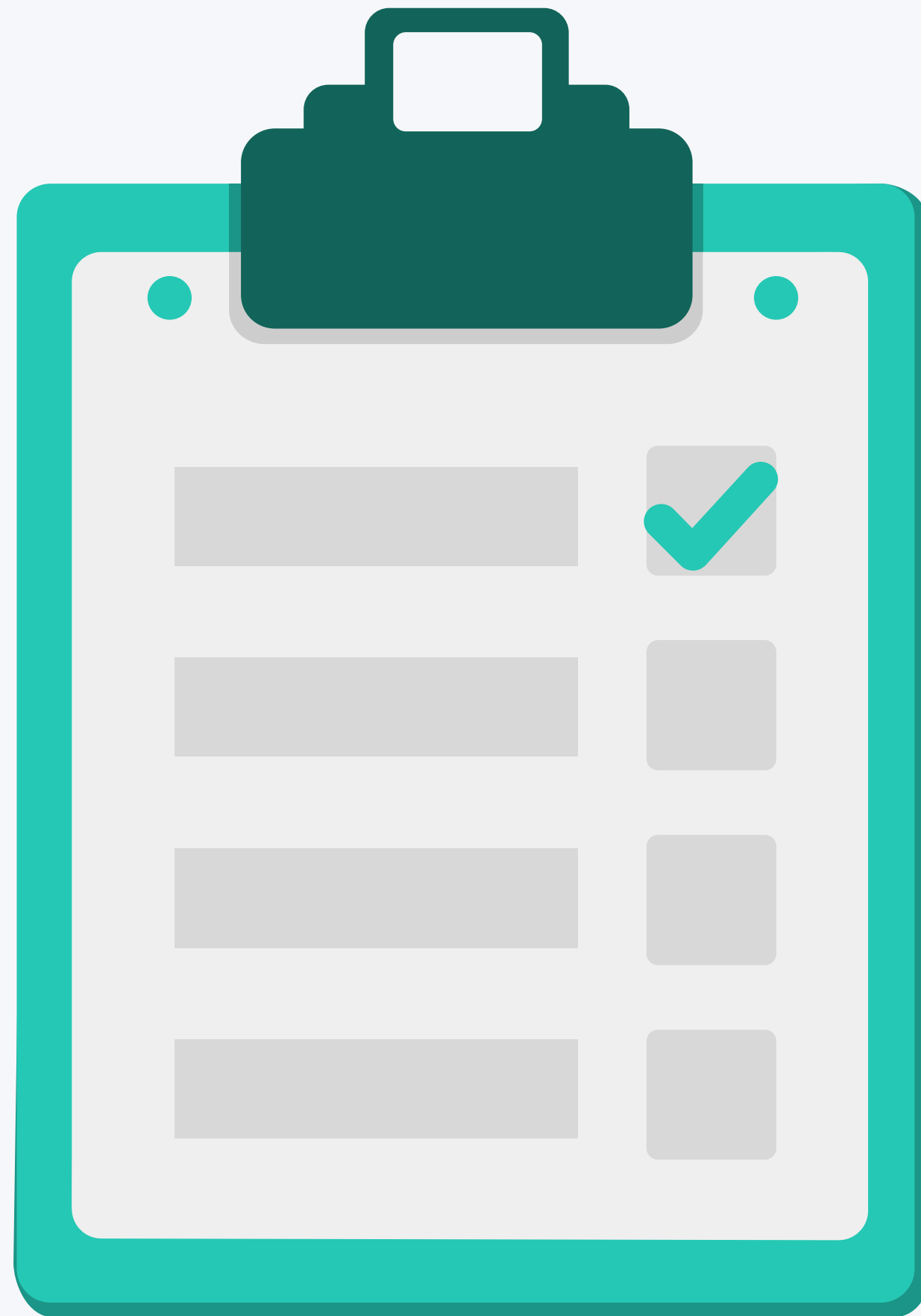
## Programmer

Contributions with AtCore, Atelier and random KDE projects.





# Our Agenda



0

Introduction.

1

Data visualization.

2

Sensor integration.

3

Control.

4

2D simulations.

5

3D simulations.



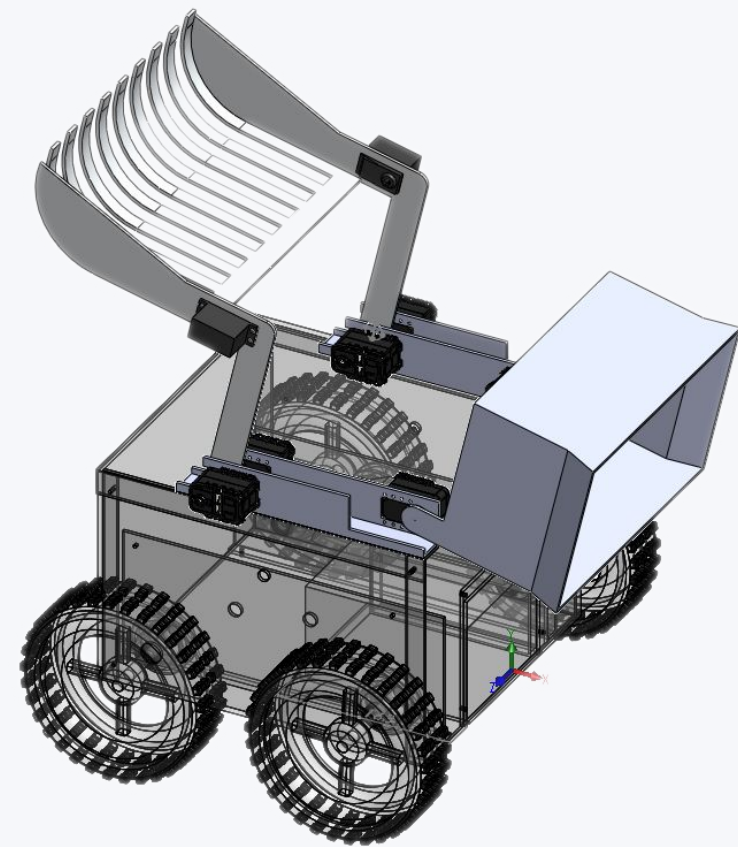
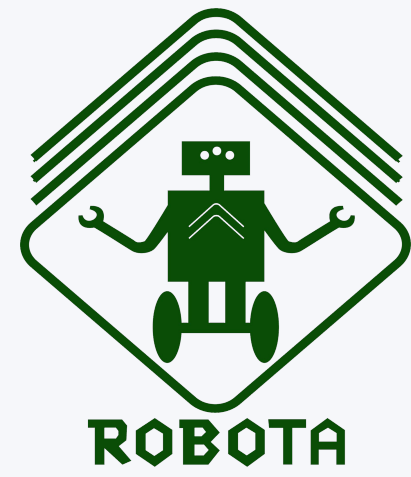
**What is a robot ?**

**Can robots be Qt ?**

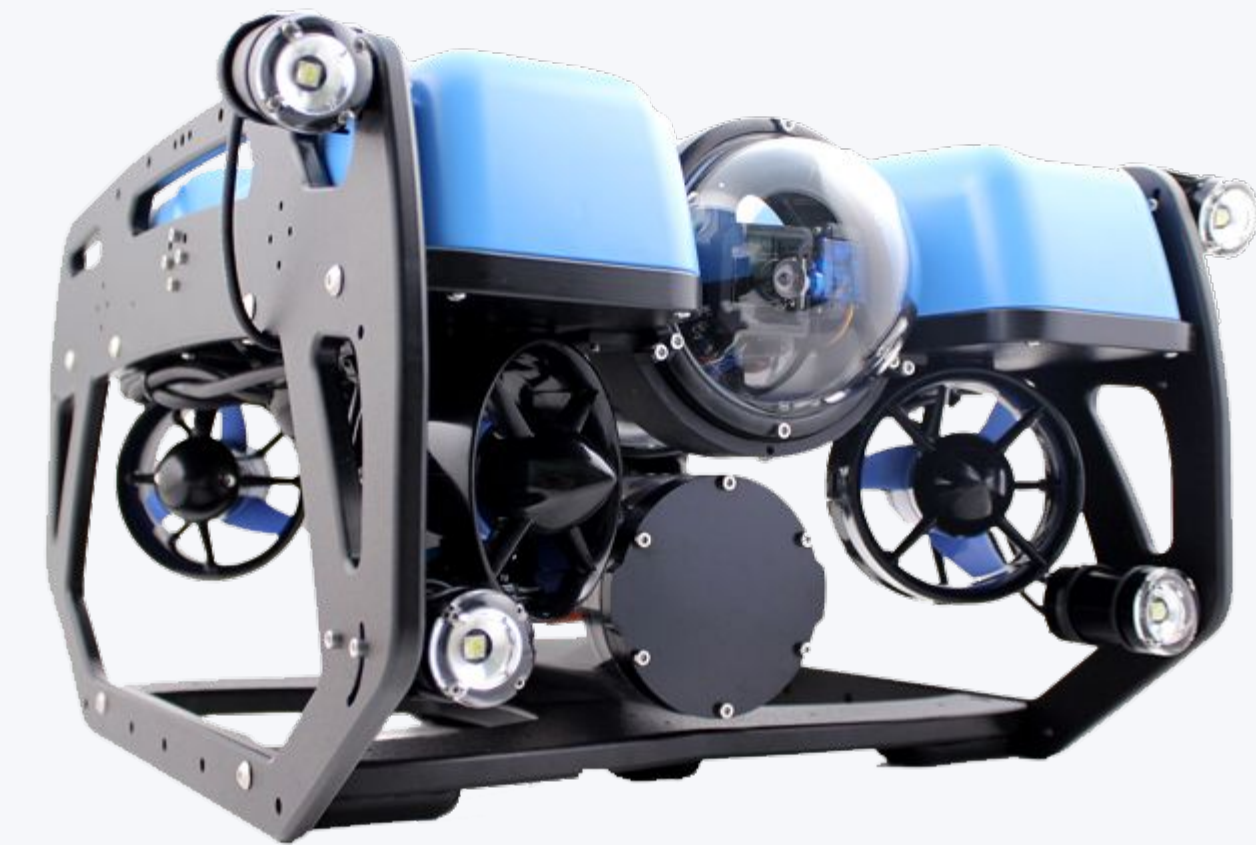




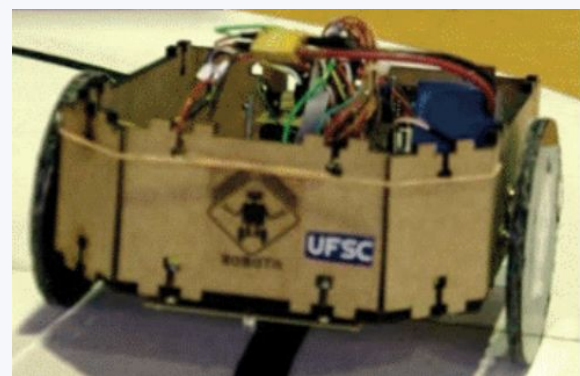
# Examples



Source: [horsaeronaves.com](http://horsaeronaves.com)



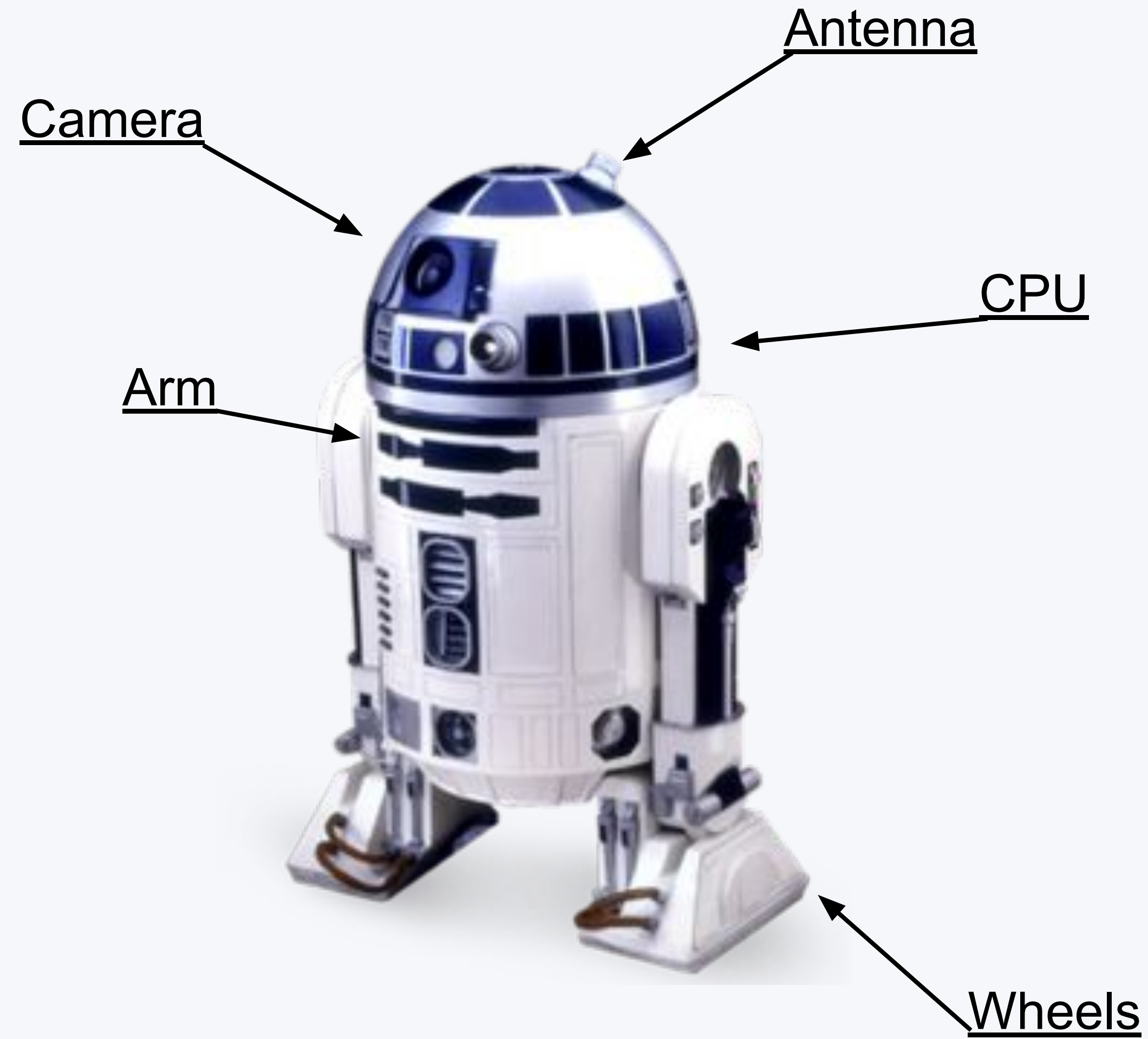
Source: [bluerobotics.com](http://bluerobotics.com)





# Show me what you got

...



# Sensors



USB



I2C



SPI



UDP



RS485



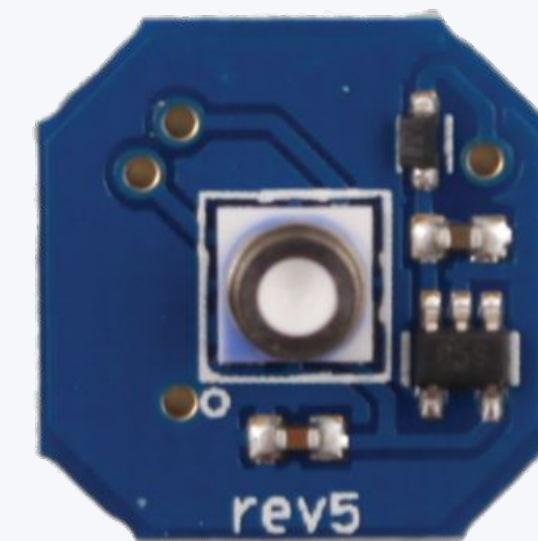
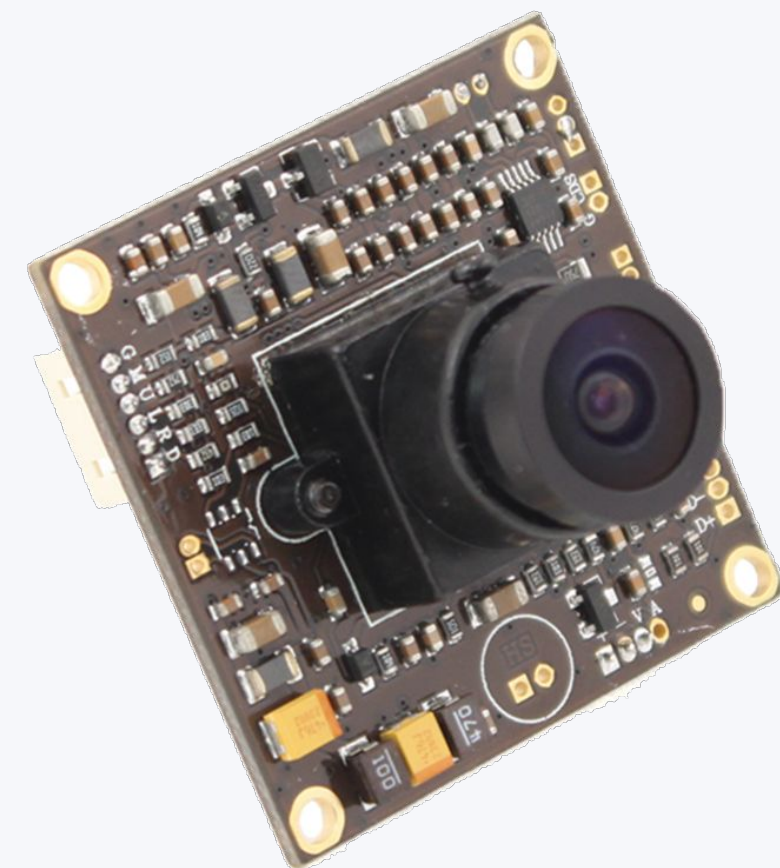
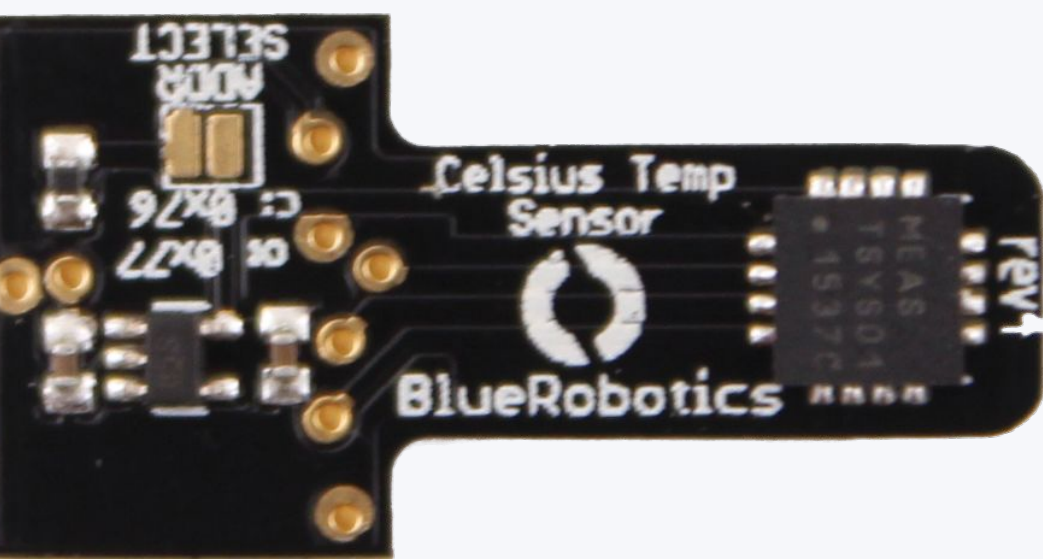
Serial



CAN



TCP





# Exposer-GUI



300 /dev/ttyUSB0 Stop

Graph

led testuint8 testuint16 testuint32 testint8 testint16 testint32 testfloat

199.0  
118.4  
37.7  
-42.9  
-123.6  
491.0 565.8 640.5 715.3 790.0

Plots

- led
- testuint8
- testuint16
- testuint32
- testint8
- testint16
- testint32
- testfloat

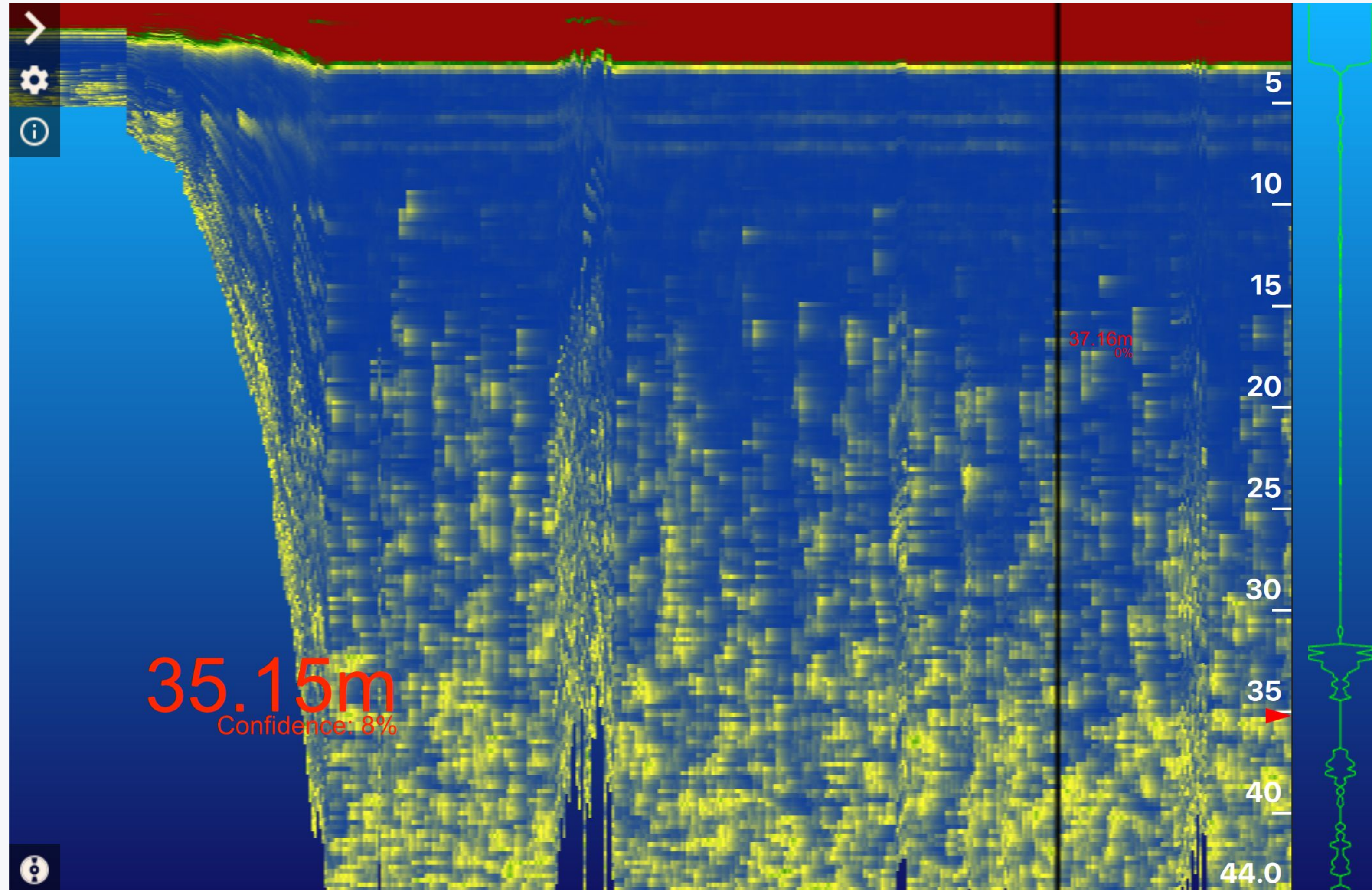
```
[17:44:51:236] <#\x06\x04\xa1\xff\xff\xff
[17:44:51:236] <#\x07\x04N"\xf7\xc2
[17:44:51:236] <#\x08\x08Robota\x20!
[17:44:51:339] <#\x00\x01\x00
[17:44:51:339] <#\x01\x01?
[17:44:51:339] <#\x02\x02\x08\x00
[17:44:51:340] <#\x03\x04\x06\x00\x00\x00
[17:44:51:340] <#\x04\x01\xdc
[17:44:51:340] <#\x05\x02\xa5\xff
[17:44:51:340] <#\x06\x04\xa3\xff\xff\xff
[17:44:51:340] <#\x07\x04N"\xf7\xc2
[17:44:51:341] <#\x08\x08Robota\x20!
[17:44:51:432] <#\x00\x01\x00
[17:44:51:432] <#\x01\x019
[17:44:51:432] <#\x02\x02\x05\x00
[17:44:51:432] <#\x03\x04\x09\x00\x00\x00
[17:44:51:433] <#\x04\x01\xd6
[17:44:51:433] <#\x05\x02\xa2\xff
[17:44:51:433] <#\x06\x04\xa6\xff\xff\xff
[17:44:51:433] <#\x07\x04N"\xf7\xc2
[17:44:51:433] <#\x08\x08Robota\x20!
```

	Name	Type	Value	Enter
1	led	uint8	0	
2	testuint8	uint8	57	
3	testuint16	uint16	5	
4	testuint32	uint32	9	
5	testint8	int8	-42	
6	testint16	int16	-94	
7	testint32	int32	-90	
8	testfloat	float	-123.567001...	
9	teststring	String	Robota !	

Operation Target [Data]: 33 0; 35 0; 34 0 1

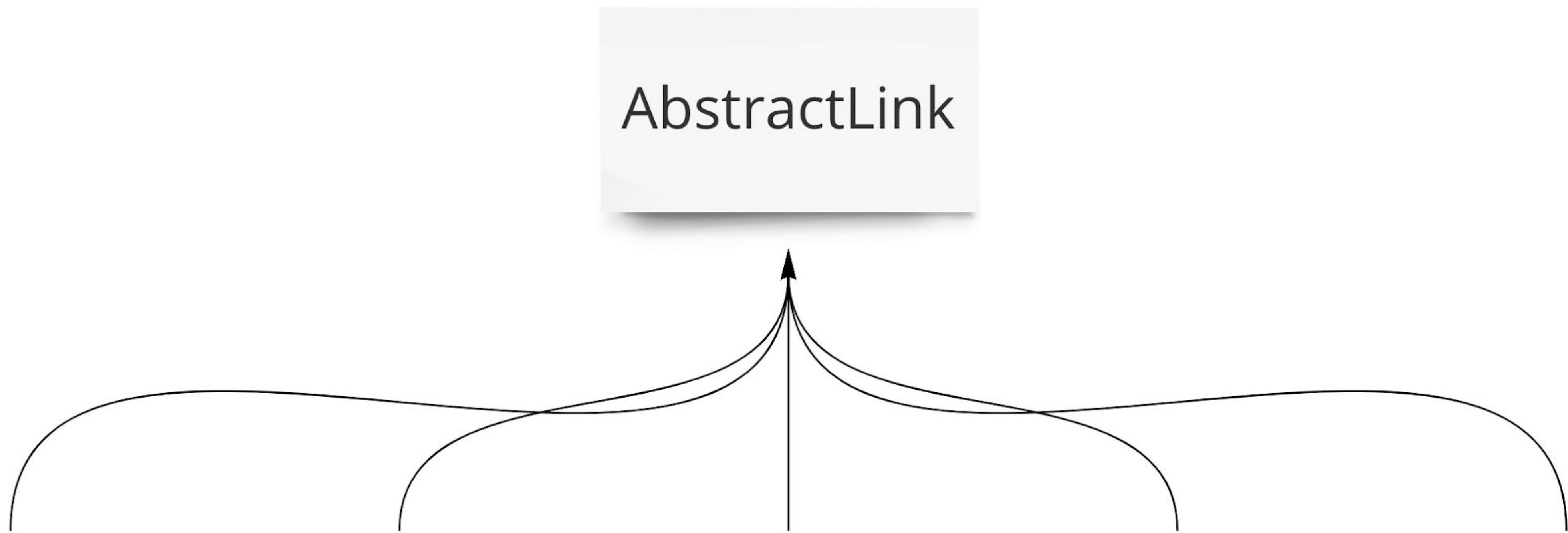


# PingViewer [Demo]





# AbstractLink



SerialLink  
(QSerialPort)

UdpLink  
(QUdpSocket)

TcpLink  
(QTcpSocket)

SimulationLink  
(QObject)

FileLink  
(QFile)



# AbstractLink

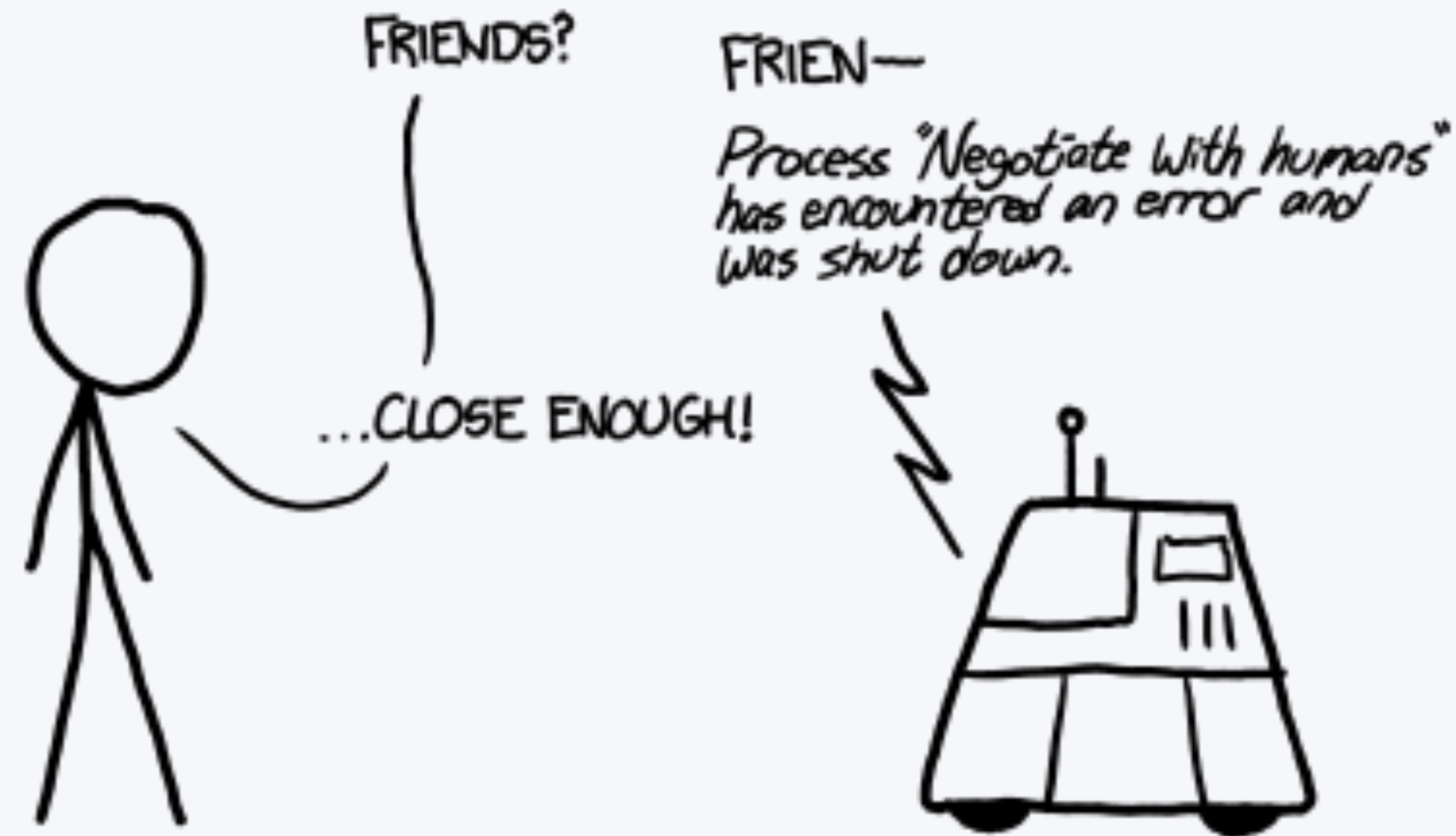


```
1. //link.cpp
2. Link::Link(LinkType linkType, QString name, QObject* parent)
3.     : QObject(parent)
4.     , _abstractLink(nullptr)
5. {
6.     switch(linkType) {
7.     case LinkType::File :
8.         _abstractLink = new FileLink();
9.         break;
10.    case LinkType::Serial :
11.        _abstractLink = new SerialLink();
12.        break;
13.    case LinkType::Udp :
14.        _abstractLink = new UDPLink();
15.        break;
16.    case LinkType::PingSimulation :
17.        _abstractLink = new PingSimulationLink();
18.        break;
19.    default :
20.        qCritical(PING_PROTOCOL_LINK) << "Link not available!";
21.        return;
22.    }
23.    ...
```

```
1. //sensor.cpp
2. ...
3. QSharedPointer<Link> _link;
4. ...
5. AbstractLink* link() { return _link.data() ? _link->self() : nullptr; };
6. Q_PROPERTY(AbstractLink* link READ link NOTIFY linkUpdate)
7. ...
8. if(!link()->isOpen())
9. ...
10. if(!link()->isWritable())
11. ...
12. if(link()->type() == LinkType::File)
13. ...
14. connect(link(), &AbstractLink::newData, _parser, &Parser::parseBuffer);
15. ...
16. SerialLink* serialLink = dynamic_cast<SerialLink*>(link());
17. ...
```



# Control







# Planes, Rovers, Boats and Submarines

---

The most advanced, full-featured and reliable open source autopilot software available. It is the only autopilot software capable of controlling any vehicle system imaginable, from conventional airplanes, multirotors, and helicopters, to boats and even submarines.









# Mission Planner



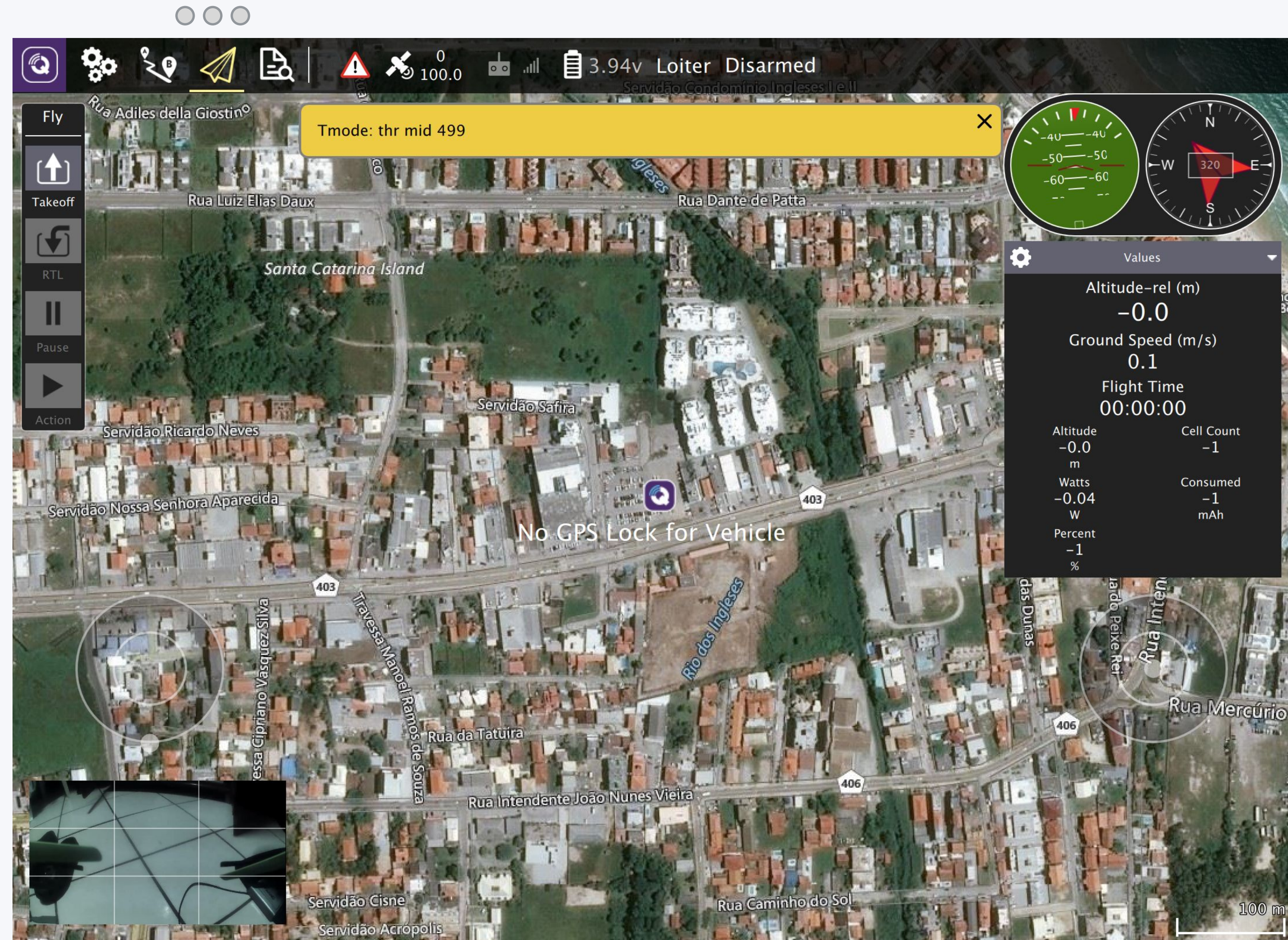
- ◆ Windows only
- ◆ C#
- ◆ Slow user input
- ◆ Older UI

	Command					Lat	Long	Alt	Delete	Up	Down	Grad %	Dist	AZ
1	WAYPOINT	0	0	0	0	-35.0407928	117.8277898	100	X	🏠	🏠	95.7	104.5	1
2	WAYPOINT	0	0	0	0	-35.0406786	117.8260410	100	X	🏠	🏠	0.0	159.7	275
3	WAYPOINT	0	0	0	0	-35.0417239	117.8251612	100	X	🏠	🏠	0.0	141.2	215
4	WAYPOINT	0	0	0	0	-35.0428395	117.8259873	100	X	🏠	🏠	0.0	145.1	149
5	WAYPOINT	0	0	0	0	-35.0427165	117.8274572	100	X	🏠	🏠	0.0	134.5	84



# QGroundcontrol [Demo]

- ◆ Windows, Linux, Mac, Android and iOS
- ◆ C++
- ◆ Faster user input
- ◆ Modernish UI

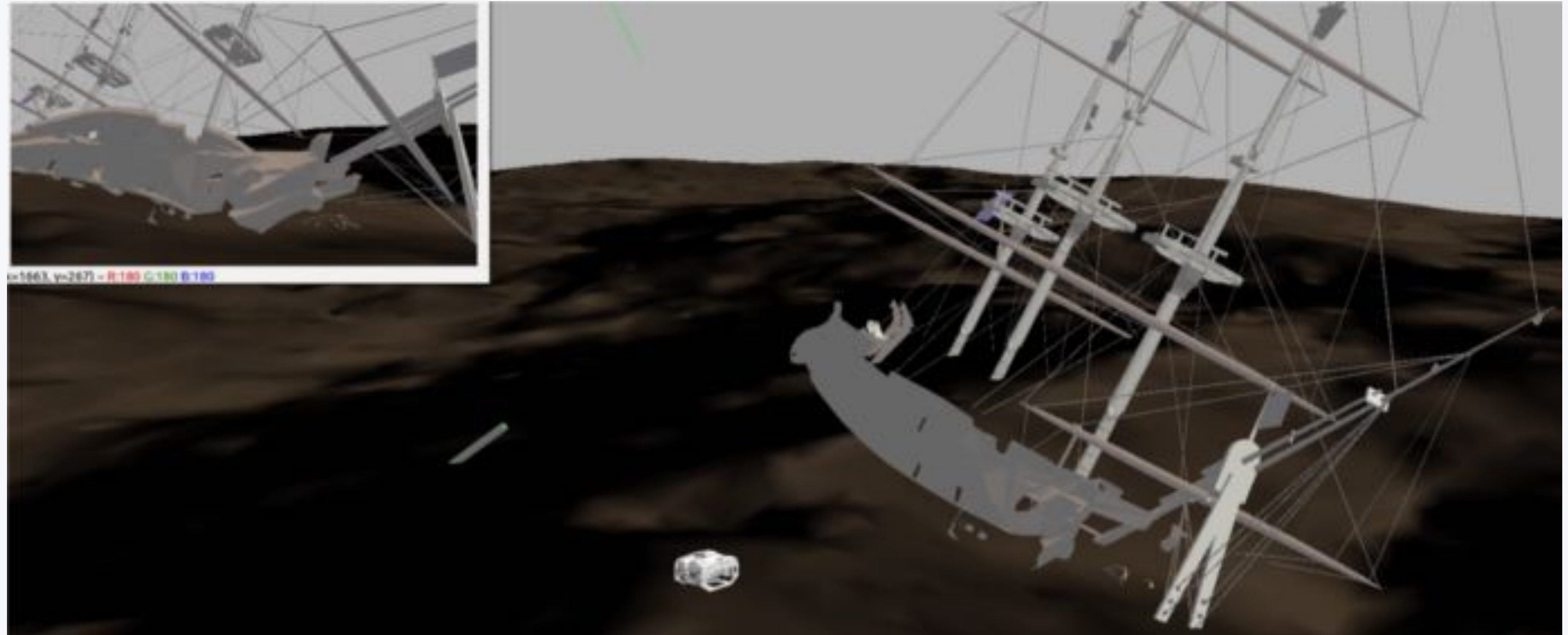




# Simulation



- ◆ Gazebo
- ◆ V-REP
- ◆ ARGoS
- ◆ Microsoft Robotics
- ◆ ...



Gazebo with BlueROV simulation



# Simulation



## Behavior

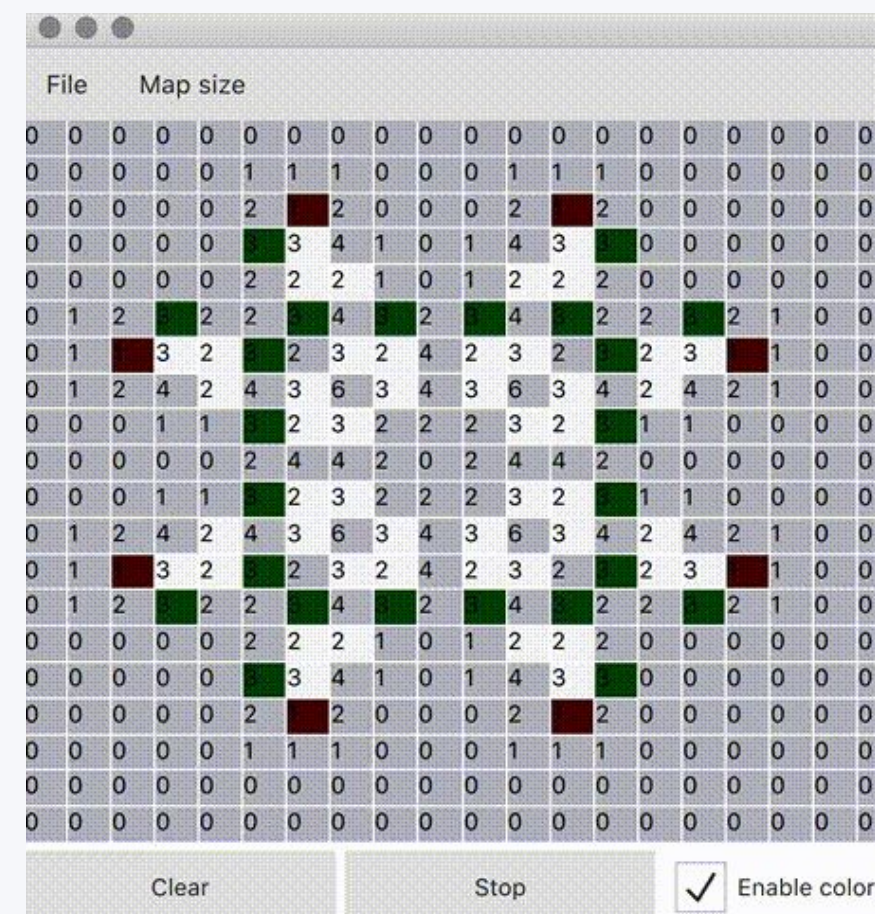
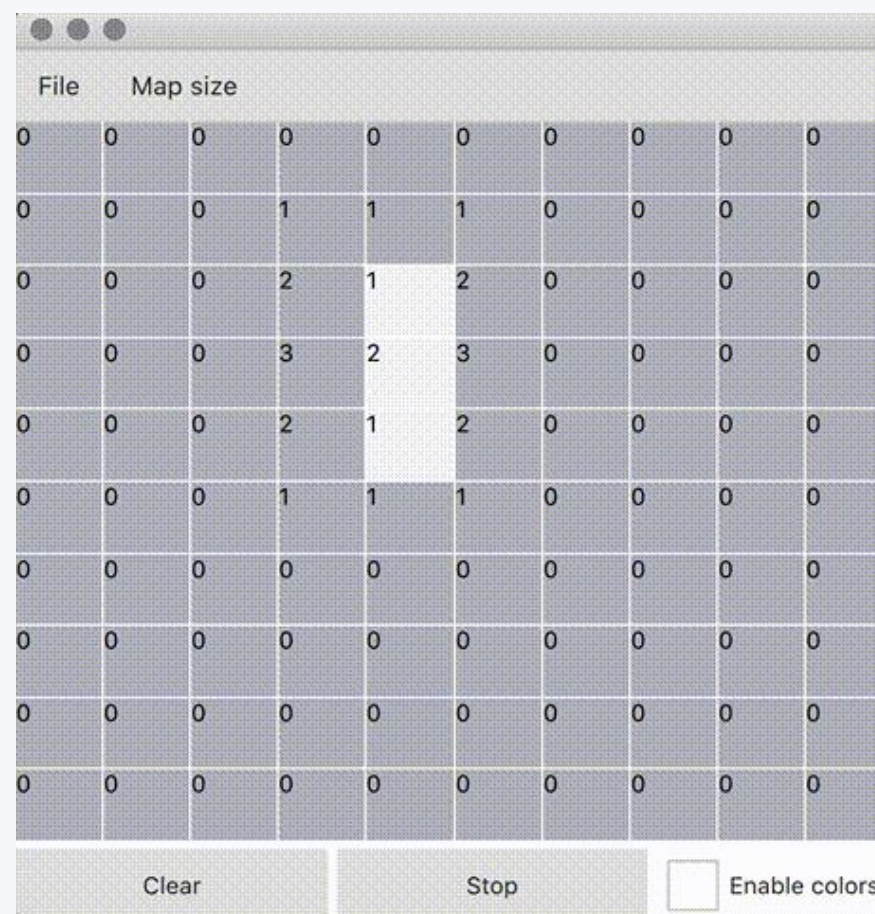
- ✓ Development
- ✓ Debug
- ✓ Validation

## Desires

- ✓ Changes based on last state.

## Interactions

- ✓ Interaction between models
- ✓ Interaction of model with ambient
- ✓ Simulation result and consequences

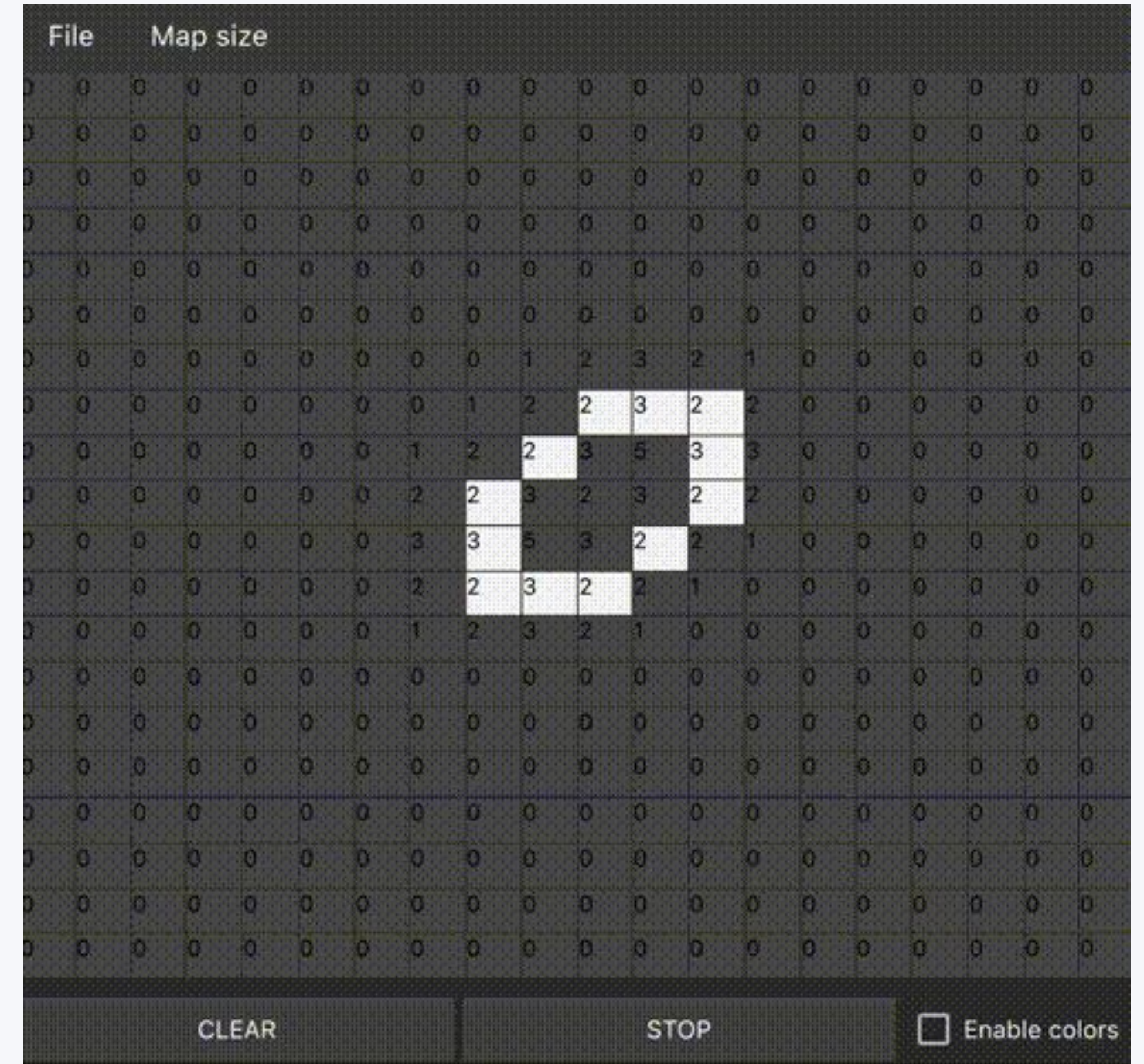




# 2D Simulation



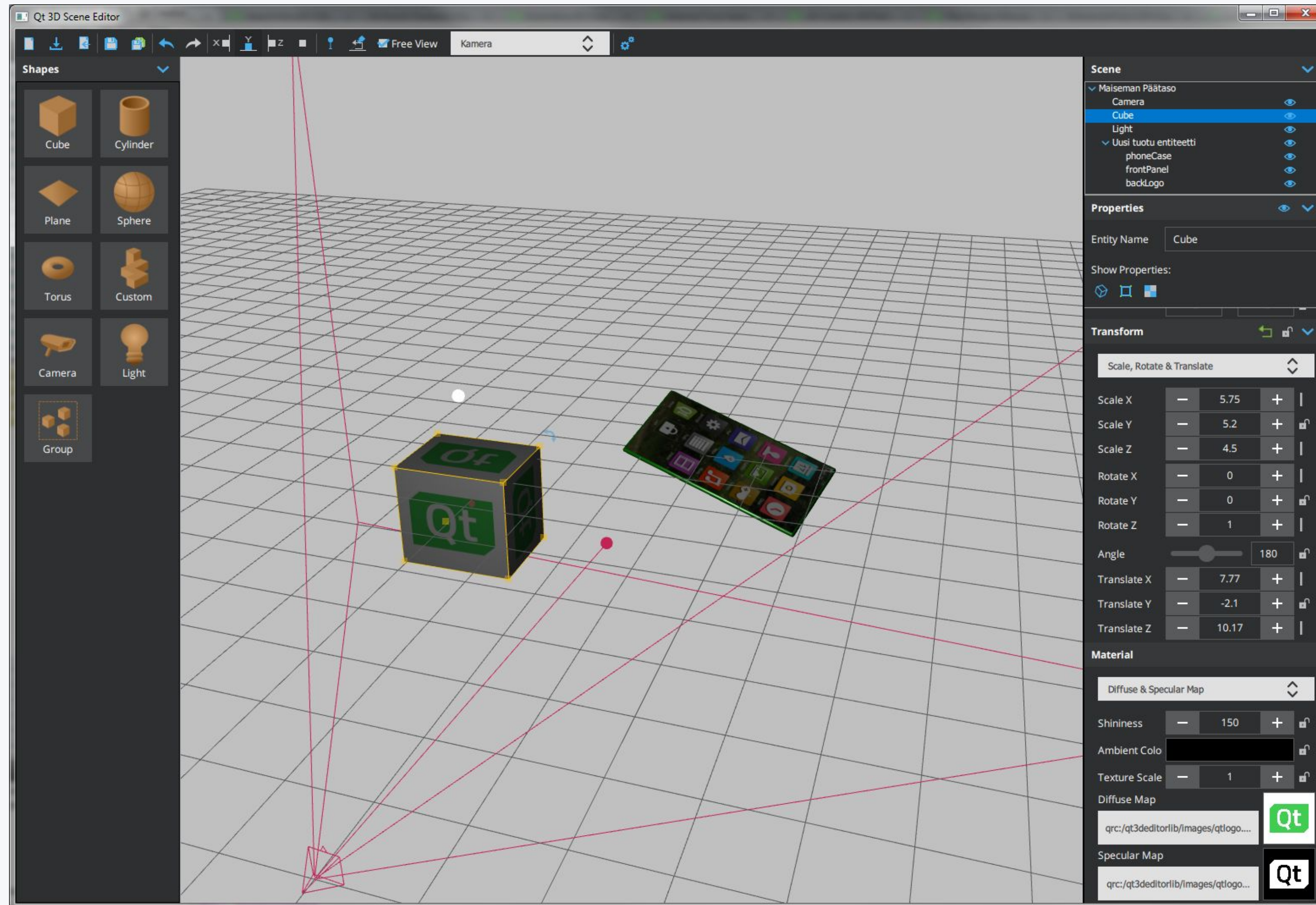
SLAM



Conway's GoL

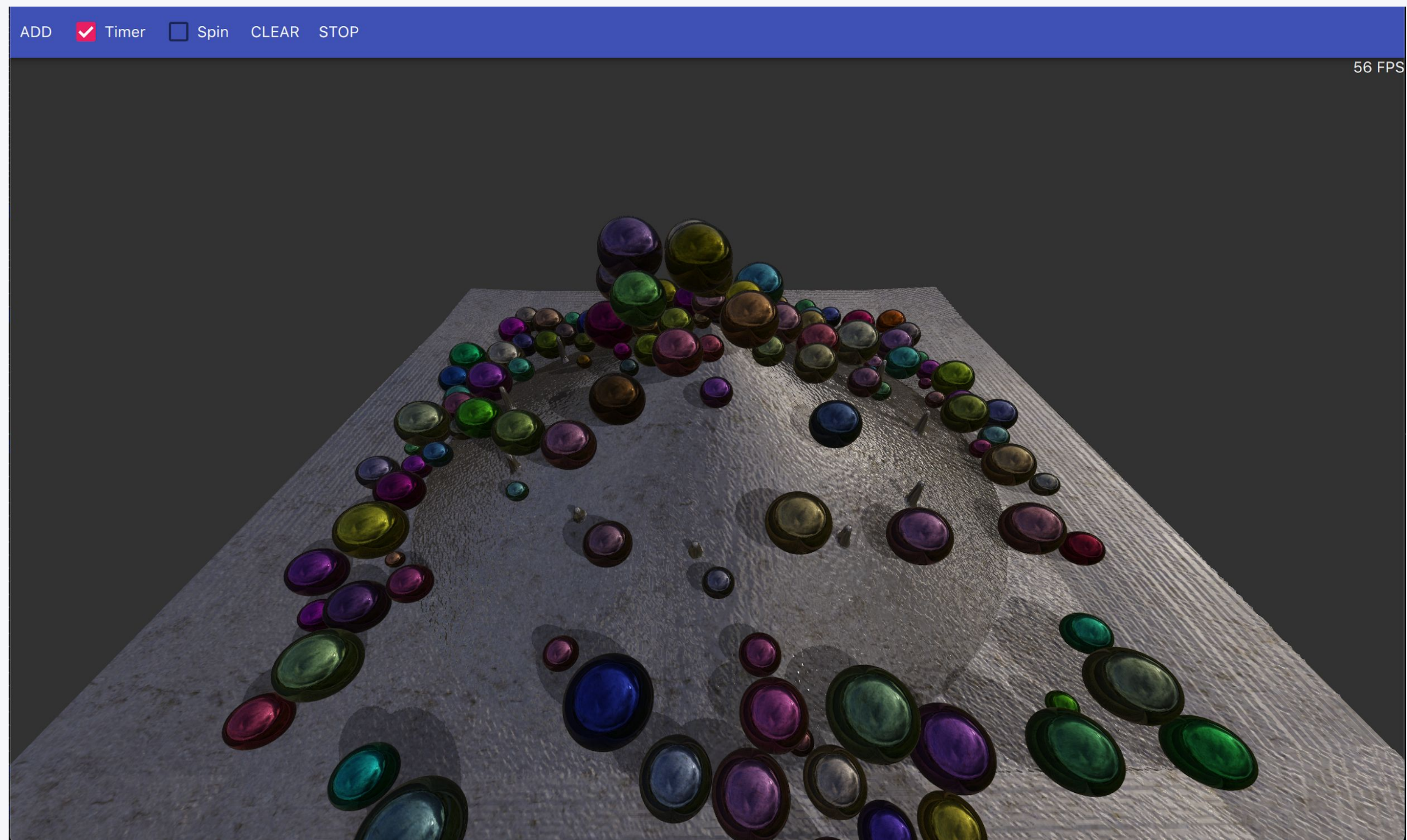


# Qt 3D





# Simple [Demo]





# Simple [Demo]



```
1. import QBullet 1.0 as Bullet
2. ...
3. Bullet.DiscreteDynamicsWorld {
4.     ...
5.     gravity: Qt.vector3d(0, -9.8, 0)
6. }
```

```
1. ...
2. BulletTools.Sandbox {
3.     heightmap: "qrc:/resources/heightmap.png"
4.     width: 250
5.     wallHeight: 50
6.     friction: 0.9
7.     restitution: 0.5
8.     ...
9. }
```



# Simple [Demo]



```
1. import QBullet 1.0 as Bullet
2. ...
3. Bullet.SphereShape {
4.   id: sphereShape
5.   ...
6. }
7.
8. Bullet.RigidBody {
9.   id: ballBody
10.  collisionShape: sphereShape
11.  ...
12. }
13.
14. Ball {
15.   matrix: ballBody.matrix
16.   ...
17. }
18. ...
```

```
1. // Ball
2. ...
3. Entity {
4.   property alias matrix: transform.matrix
5.
6.   Material { id: material }
7.   SphereMesh { id: mesh }
8.   Transform { id: transform }
9.
10.  components: [ material, mesh, transform ]
11. }
12.
```



# Complex [Demo]





Qt



Obrigado!

perguntas?



QtCon  
BR



Qt



@patrickelectric

ArduPilot/ardupilot

bluerobotics/ping-viewer

bluerobotics/ping-protocol

mavlink/qgroundcontrol

patrickelectric/Conway-game-of-life-in-pyside2

patrickelectric/Qt3D-Konqi-Simulator

patrickelectric/SLAM\_Qt

robotadasufsc/Exposer

robotadasufsc/Exposer-Gui

Williangalvani/provant-groundstation



csaga77/bullet-physics-qml-plugin



@patrickelectric



@patrickelectric



<https://patrickelectric.work>

**Patrick Pereira**

patrick@bluerobotics.com

patrickjp@kde.org